NPS-52Pw76101

NAVAL POSTGRADUATE SCHOOL

Monterey, California

ASPECTS OF MICROCOMPUTER STANDARDS

V. Michael Powers

28 October 1976

Prepared for:
Naval Electronics Laboratory Center
271 Catalina Blvd.
San Diego, CA 92152

NAVAL POSTGRADUATE SCHOOL
Monterey, California

Rear Admiral Isham Linder
Superintendent

Jack R. Borsting
Provost

ASPECTS OF MICROCOMPUTER STANDARDS

This is the final report of a project funded by the Naval Electronics Laboratory Center under Work Request Number N5376WR00059.

This report was prepared by:

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: NPS-52Pw76101	2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle): Aspects of Microcomputer Standards	5. TYPE OF REPORT & PERIOD COVERED: Final FY76
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): V. Michael Powers	8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Code 52Pw Naval Postgraduate School Monterey, CA 93940	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: NELC Work Request N0095376 WR00059
11. CONTROLLING OFFICE NAME AND ADDRESS: Code 4300, NELC San Diego, CA 92152	12. REPORT DATE: 28 October 1976
	13. NUMBER OF PAGES: 16
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report): UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

microcomputers
standards
Navy microcomputers
tactical microcomputers
microprocessors
high level languages

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The Navy must have standards to aid designers in their choice of microcomputer designs. High-level applications and system programming languages have several advantages in microcomputer design for a wide variety of applications.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

1. Introduction

Most of the electronic systems to be developed from this point on, for the Navy as for the country in general, will be amenable to inclusion of microcomputers. In view of the profusion of different hardware and software components currently in use in microcomputer development, it is appropriate that the Navy attempt to protect itself from intolerable life cycle support problems and costs by means of establishing standards for microcomputers. In the Navy, a standard can mean two things. First, it can mean a level of quality and performance in specifications, design documentation and construction. Second, it can in some cases mean the choice of a uniform set of components to be used over a wide range of applications. Either interpretation or both may be appropriate for Navy microcomputer standards.

This report grew out of a brief participation with the Naval Electronics Center in an effort to prepare input for the eventual formulation of a Navy microcomputer standard. Their portion of the task resulted in a position paper [4], which was also the foundation of a public presentation [3]. The NELC paper was written after meeting with representatives of several Navy organizations, but did not necessarily reflect a consensus opinion.

The objective of our portion of the task has been to comment on some of the issues which seem most important in producing a standard which is reasonable with respect to the present technology and its direction and rate of progress. Some of these comments have been previously transmitted to NELC as working papers and notes, and some have been presented at working meetings.

This report contains a collection of comments, including some which have been previously shared with the other task participants, and even including some which are in substantial agreement with statements in the NELC position paper [4]. The intent here has been to emphasize aspects of the problem which are particularly important, which were not strongly stated or clearly resolved in the position paper, or which generate opinions which may differ from the position paper.

During the period of this task, some of the conversations with other members of the effort, and other experiences, have highlighted some long-range problems with the use of microcomputers in Navy systems. Suggestions for future work on these topics are included in the report.

2. Cautions

The main purpose in establishing a standard for microcomputers is to facilitate effective and economical choices. Assuming that a designer is considering the use of microcomputers to implement his functions, an effective statement of a standard will identify the general trend of selections in his community. He should realize that deviations from or exceptions to the standards may involve life cycle or indirect costs, extending beyond his immediate concerns, which will be significant, and which will impair the usefulness of his product.

In order to realize the purpose of the standard, it is NOT necessary to prevent designers from taking advantage of developments in the state-of-the- which are needed to economically support his application. Proper formulation and use of a standard will foster better, not less, application of complicated technology with the best interests of the total context of the application served as efficiently as possible.

It is characteristic of microcomputers that advances in the state of the art, particularly in manufacturing technology and device marketing, are still proceeding almost faster than users can comprehend. With monthly important developments the expected norm in the near future, it does not make sense to specify any given hardware product as the only permissible unit to use from here on. Choosing a single product as "the standard" only makes sense for a short term, and then only if escape mechanisms are provided.

A standard for a technology which is so highly volatile, in order to maintain its effectiveness for more than a few months, must allow for new developments and changes. There are only two ways to do this. One is to phrase the statement of the standard at a level higher than the level of the changing devices. The other is to provide a mechanism for growth, through updates to the standard, through the granting of exceptions, or both.

It is to be expected that some designers will find it difficult to work effectively within the guidelines, no matter what they are. Unforeseen circumstances, primarily the rapid changes expected in the state of the art, will render the standard inappropriate for some cases. Appeals against the standard should be quickly evaluated from a technical standpoint, both to review the need to deviate from the prescribed standard and to accumulate date on which to base continued updating of the standard. Appeals and their decisions could be published within the design community.

The "no standard" option, however, neither protects the Navy from the unmanaged proliferation of incompatible designs nor offers the designer any helpful guidance.

3. Applications

Microcomputers are generally thought to be appropriate for a range of applications, from some previously implemented in random logic to some previously implemented with minicomputers. The low cost of microcomputers is increasingly leading to applications which would not have been attempted by either previous means. The area of tactical systems is not expected to have much less a variety of applications than the commercial world. Human interfaces, dedicated control, communications processing, simulation, data storage and retrieval, numerical calculations, image processing and display, are all potential microcomputer applications areas.

Perhaps there should be separate statements of standards to be used by the (possibly different) designers working at the different levels. For hardware, software, total system design and support system it must be separately decided according to which of the following levels the standard should best be formulated:

Function

Organizational pattern (logic)

Performance

Structure

Implementation

It may be dangerous to characterize the computing resources and features required by a community according to the subject matter upon which they work. In the early days of hardware development, the device now known as an index register was thought to be appropriate for data processing but not for "scientific" work. Until the number crunchers found this was just what they needed!

A modern example might be the tendency to generalize and say that, for example, signal processing requires something special in the way of processor speed. Perhaps this is true in some cases, but more may be gained by investigating how the cheap, unified-architecture processors can be applied than by only admitting microprogrammed, multi-chip structures to the users who are concerned with signal processing.

As examples, consider the oscilloscopes which use microprocessors for housekeeping, calibration, and switching among analog sections. Or consider the following, from the abstract of a recent student thesis from NPS:

"...This thesis establishes that an 8-bit [single-chip] microprocessor, interfaced to a common surface search radar, could compute the course and speed of multiple radar targets. The investigation includes... the development of a Kalman filter algorithm...and the microprocessor system software [6]."

4. High-Level Languages

As noted above, focussing on the design level, rather than on the chip implementation level, offers the possibilitity of formulating a standard which contributes to the effective use of microcomputers, but avoids being outdated by the marketing of new chips. As discussed below, a high level compiler-like language is envisioned as the implementation of a direct design level. The source code serves as a widely readable design language and contributes greatly to the documentation of the design. What follows is a brief summary of the major issues.

4.1. Efficiency

The basic fact here is that most examples of a very small program,

carefully coded once in assembly language and once in a high level language, show that the compiler generates more instruction words. This means that the compiler's version requires more program storage than the hand-coded version. In some cases the compiled version includes some redundant code, such as a redundant 'halt' sequence or predefined utility routines which are not used in the particular program. These redundancies can be minimized by careful construction of the compiler. 'Optimizing' compilers attempt to make not the most obvious but the most economical mapping from source to object code versions of the program. Most of the current discussion of efficiency concentrates on the number of bytes of instruction storage required to implement a given algorithm. There may be a weak correlation with the execution time. An 'efficient' compiler may be expected to produce code that is in the neighborhood of 20% longer than a carefully coded small assembly language program.

Designers in the microcomputer field are beginning to realize that there are two misleading aspects to such a discussion of efficiency. In the first place, designing in a high level language is much more efficient in the overall project, in view of the tremendous advantage in readability possessed by high-level languages for designing, debugging, documenting and maintaining the product. In the second place, there is little evidence that the advantage in efficiency is retained by assembly-language programmers in large programming efforts. There are programs operating today which would never have been attempted in assembly language because of their large size. One example is Reference [6] cited in the preceding section.

Programmer productivity is sometimes discussed in terms of average number of lines of debugged code generated per day. This measure seems

to be independent of whether the programmer is coding in a very low level or in a high level language. Considering that many low level statements are required to do the work of one high level statement, it appears that a high level language is a tool to greatly increase productivity. It is also much easier to maintain (fix) an old program written in a high level language.

4.2. Software Compatibility

While it may very well be true that compatibility of microcomputer designs with present "big computer" programs is not required, it is important that designers be able to easily transport design units (routines) from microcomputers of one manufacturer to another, between differently configured microcomputers, and to new processors of presently unknown instruction format. The PL/M compiler, for example, was originally written to produce code only for the 8008 processor. When the 8008 was introduced, and the compiler modified to produce code from the new set of instructions, it was simply a matter of recompiling to capture any of the 8008 software, written in PL/M, which was worth keeping.

It has been suggested that Navy use of microcomputers be linked with an attempt to use compilers for languages such as CMS-2 and FORTRAN, in order to "capture existing software." Despite the valid concern with the cost of rewriting software, there are two fallacies in the reasoning behind this suggestion. One is that microcomputers, in the way they exploit their cost advantage over random logic, tend to have a different way of organizing the task - the old programs are not right for the job. Those portions of the old software which are still appropriate, purely mathematical computations, for example, can be recoded relatively easily. The difficult

and costly portions of the software conversion effort involve mostly the specifications, system logic and interfaces, and device control. These are the portions which must be rewritten anyway for the new architecture.

The second fallacy in the reasoning is in ignoring the maintainance cost of old software. If a program costs each year nearly a third of the development rate of cost just for maintainance, how many times are we to continue to buy that old program over and over?

4.3. Features

More than one high-level language has been used for microcomputer program development. It has been found, for example, that BASIC can be eminently satisfactory for applications devoted to numerical calculations, such as those involved in celestial navigation. Other software, however, such as resident language processors, operating system components and direct device control programs require a system programming language.

At least in the case of dedicated control applications, the software must allow the designer to invoke operations (bit manipulations, for example) which are done directly by the hardware.

It appears necessary that such a system programming language be a block-structured language. The features which appear necessary include:

* macro capability, to improve readability
* procedures and program blocks in which variables have limited scope
* the common programming structures WHILE, IF-THEN-ELSE, CASE or SWITCH, and an iteration loop
* predefined primitive machine operations such as ROR (rotate right), masking, I/O

Of course, the present leader in the market, PL/M, has these features.

Convenient but not necessary features include the production of relocatable code (removes the programmer one more step from the machine level and requires a more complicated resident loader) and subroutine libraries (most effective when separately compilable into relocatable code).

Features which do not seem suitable for microcomputer development are extensibility by the programmer and in-line assembly language.

5. Future Work

5.1. Design Languages

It is no contradiction that the appearance of cheap hardware (microcomputers) is a call for more work in software. The best technological hope for an efficient and cost-effective approach to the employment of microcomputers lies with the development of better system generation, documentation, test and maintainance software tools. Effective design and management of the explosion of applications of microcomputers in diverse applications requires that we have tools which allow us to work at a sufficiently high level of abstraction. The human mind has a limited attention span, and a limited capacity in the number of details which can be clearly recalled. Software design must have tools which allow the suppression (at least temporarily) of details belonging to a lower level. Recent developments in software which have helped software designers to work at a more effective, less cluttered level include the early ideas of modularity, subroutining, block structured programming, top-down design and the principles of "information hiding."

It is possible that we have not reached the ultimate in abstracting the important structural facets of a software design. Perhaps there are other media for software design than the traditional programming languages with their blank verse appearance and stilted syntax.

One possibility is a digital designer's familiar tool in a new light. It has been found in recent work at NPS that least in one problem domain state diagrams are a very effective program design medium. In an attempt to produce software for the microcomputer implementation of a telecommunication message protocol, it has been found that description of the algorithm to support this protocol is superior to English text description, flow charts or programming languages. Perhaps this advantage is related to the popularity of decision tables, a specially stylized form of state tables, in some parts of applications.

It appears that state descriptions of protocols can be as free from ambiguities and as amenable to automatic translation as the conventional programming languages. More work needs to be done to establish methods and domains of applicability of this method of software design.

Whatever the medium of expression of design, the variations of architecture beginning to appear in microprocessor products also demands renewed examination of means of "automatic programming." Bit-slice computers are usually microprogrammed. Computers from some manufacturers (ATMAC from RCA and MAP from CSPI, for example) separate the address and control computation from the data operators. Further, it appears reasonable that some computers in the future will be composed of a control structure coordinating an array of data operators, whose number and types have been chosen specifically for a particular application. Both microprogrammed and split-function computers deserve compiler code generation tailored to their

peculiar attributes. Indeed, the addressing of some of the microprogram structures available today is so complicated that it may require automatic means to use effectively [1].

5.2. Interfaces

The interfaces between components of a system will continue to be one of the major difficulties in Navy systems. Significant improvements can be obtained by incorporating microcomputers in the major interfaces of tactical systems; it should be possible to design "smarter" interfaces, and interfaces which can serve more varied equipment configurations with only software changes.

Part of the standards effort, however, should protect against the possibility that proliferation of microcomputers could bring a multiplying proliferation of problems with interfacing between and among the microcomputer components.

Intracomputer interfaces deal primarily with the computer "bus" structures which interconnect the processor, memory and input/output devices. A standard or a design specification for a computer bus would specify the signal lines, their electrical and logical characteristics, possibly mechanical connector details, and certainly the protocol or detailed sequence of logical actions for communicating on the bus. Economy of design would seem to point to several different busses for the several styles of computer architecture. There seems to be a tendency, however, among the producers of development systems for the popular commercial microcomputer lines, to develop similar bus structures. Such a tendency may with some work be developed into a standard. Present indications

are that such a standard would differ significantly from IEEE Std 488-1975, one of which has been mentioned as a possibility but which originated to serve a quite different purpose.

Another interface which deserves more attention is the connection among data stations, many of which may be or may contain microcomputers. Inter-microcomputer communications should be subject to (perhaps plural) standards, but more attention is needed to this topic. It has not yet been established which forms of communication are appropriate. Several promising approaches for different circumstances are SDMS, MIL STD 1553 and modified ring structures [2].

A topic which deserves more attention and which may ultimately be more important than the simple choice of intracomputer or intercomputer connection standards is the development of an improved way of specifying the interfaces. One somewhat formalized method of specifying communication interfaces is under development at NPS. This approach deserves more attention, to determine its applicability to the specification of linkages between microcomputer units. It may in fact prove feasible to specify an interface in such a way that implementation of a standard interface becomes mostly a matter of directly compiling the standard protocol for the particular configuration.

6. Summary

The effort to gather opinions of the Navy users as a base for the formulation of a standard for tactical system use of microcomputers has, at the very least, brought out some of the differences and similarities of opinion. The NPS participation in the task has been to contribute to the

formulation and discussion of the issues by informal means. This report may be considered a postscript to that effort.

It appears obvious that low-cost microcomputer hardware of unprecedented power, complication and nearly universal applicability will be used in most Navy electronic systems in the future. The most critical requirement, therefore, is that we do what we can to assist the orderly development of the means to handle this equipment lest the purchasers of low-cost hardware drown in the cost of specifying, designing, altering, testing, procuring, stocking and repairing the pieces and their interconnections. The most critical aspects of microcomputer systems are their software.

Two areas of future work, which because of the work done under this task appear to be most promising, are continued examination of the possibilities for higher levels of system design languages, and development of experience with and tools for interconnection interfaces.

REFERENCES

1. "Intel series 3000 reference manual," Intel Corp., 1976.

2. J. Q. Jackson, "Shipboard application of a ring structured distributed computing system," Naval Postgraduate School Master's Thesis, June 1976.

3. R. Martinez, R. Peterson, "An approach to microprocessor/minicomputer standardization in Navy systems, WESCON 76 Session 11, September 15, 1976.

4. NELC, "Position paper on microprocessor/microcomputer standardization," Code 4300, Naval Electronics Laboratory Center, San Diego, 17 December 1975.

5. V. M. Powers, "A navigation microcomputer and shipboard information distribution," NPS-72Pw76011, Naval Postgraduate School, 7 January 1976.

6. C. H. Wilson, "Surface search radar tracking by a microcomputer Kalman filter," Naval Postgraduate School Master's Thesis, June 1976.

INITIAL DISTRIBUTION LIST

Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
Library Naval Postgraduate School Monterey, California 93940	1
Dr. Ralph Martinez, Code 4300 Naval Electronics Laboratory Center San Diego, CA 92152	2
Mr. William Norr (Code 2031) Naval Air Development Center Warminster, PA 18974	1
Dean of Research Naval Postgraduate School Monterey, California 93940	1
Chairman, Computer Science Department Code 52 Naval Postgraduate School Monterey, California 93940	1
Chairman, Department of Electrical Engineering Code 62 Naval Postgraduate School Monterey, California 93940	1
Prof. V. Michael Powers Code 52Pw Naval Postgraduate School Monterey, California 93940	5

[redacted]